Data Analysis and Visualization Exercise 11&12

Daniela Klaproth-Andrade, Felix Brechtmann, Julien Gagneur

Section 00 - Getting ready

1. Make sure you have already installed and loaded the following libraries:

```
library(ggplot2)
library(data.table)
library(magrittr)
library(tidyr)
library(ggrepel)

library(caret)
library(plotROC)

library(randomForest)
library(rpart)
```

Section 01 - Logistic regression on Diabetes dataset

In this section we are considering the dataset `pima-indians-diabetes.csv` which is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. A more detailed description of the data can be obtained from Kaggle: https://www.kaggle.com/uciml/pima-indians-diabetes-database.

Load the dataset with the following lines of code:

```
diabetes_dt <- fread("extdata/pima-indians-diabetes.csv")
diabetes_dt[, Outcome := as.factor(Outcome)]

# Store feature variables that we will need for later
feature_vars <- colnames(diabetes_dt[,-c("Outcome")])

diabetes_dt
```

1. Is the diabetes dataset balanced?

2. Create an appropriate plot to visualize the relationship between the `Outcome` variable and the feature variables `Glucose`, `BloodPressure` and `Insulin`. What do you conclude from your visualization?

3. Fit a logistic regression model for predicting `Outcome` only based on the feature `Glucose`. Inspect the coefficients of the model's predictors. What do these coefficients mean?

4. Create two further logistic regression models for predicting `Outcome`. For one model, use only the feature variable `BloodPressure` for building the model. For the other model, use only the feature variable `Insulin`. Which models have a significant feature?

5. Collect the predictions of each model for all samples in the dataset. Store the scores of each model in a separate column of the original dataset. Visualize the distributions of the scores with an appropriate plot.

Which type of distribution would you ideally expect? Hint: Use the `predict()` function.

6. Now, create a function for computing the confusion matrix based on the predicted scores of a model and the actual outcome. The function takes as input a threshold, a data table, the name of a scores column and the name of column with the actual labels. Then, use the implemented function for computing the confusion matrix of the first model for the thresholds -1, 0 and 1. Are there any differences? What is the amount of false positives for the last cutoff? You can use the following definition of the function:

```
confusion_matrix <- function(dt, score_column, labels_column, threshold){ }
```

7. Use the implemented function to create a second function for this time computing the TPR and FPR for a certain threshold of a classification model given the predicted scores of a model and the actual outcome. What is the TPR and the FPR of the first model for the thresholds -1, 0 and 1? Plot these values in a scatter plot. Your function should take the same parameters as before and return a data table as follows:

```
tpr_fpr <- function(dt, score_column, labels_column, threshold){
  tpr <- NULL # TODO
  fpr <- NULL # TODO
  return(data.table(tpr=tpr, fpr=fpr, t=threshold))
}
```

8. For a systematic comparison of the previously built three models, plot a ROC curve for each model into a single plot using the function `geom_roc` from the library `plotROC`. Add the area under the curve (AUC) to the plot. Which is the best model according to the AUC?

9. Now, fit a logistic regression model with all feature variables (stored in `feature_vars`). Visualize the distribution of the predicted scores for positive and negative classes. What can you conclude from this visualization regarding the separation of the two classes by the model? Plot once again the previous ROC curves and include the ROC curve of the full model for comparison.

Section 02 - Random Forests on Diabetes Dataset

1. Build a decision tree using the `rpart` function from the library `rpart` for predicting the `Outcome` given all feature variables. You can use the following command for this:

```
dt_classifier <- rpart(full_formula,
                       data =diabetes_dt,
                       control = rpart.control(minsplit = 3, cp = 0.001))
diabetes_dt
```

Note that `cp` determines when the splitting up of the decision tree stops and `minsplit` determines the minimum amount of observations in a leaf of the tree.

2. Plot a ROC curve for the decision tree. What do you observe?

3. Build a second decision tree model this time using a train-test split strategy. This means that you will use 70% of the data for training and 30% of the data for testing. Plot the ROC curves for the performance on the training and on the test dataset. What do you conclude from this?

4. In the lecture we learned that random forests are more robust to overfitting. Build a random forest using the `randomForest` function from the library `randomForest` for predicting the Outcome given all feature variables using the same train-test split strategy from before. Set the following values for the following hyper-parameters:

- `ntree = 200` for the number of trees in the forest,
- `nodesize = 20` for the maximum amount of leaf nodes,
- `maxnodes = 7` for the minimum size of leaf nodes and

- `mtry = 5` for the number of variables randomly sampled as candidates at each split.

Plot the ROC curves for test and train set for the build random forest.

5. [OPTIONAL] Try changing the hyper-parameters with the aim of achieving a better performance on the test set evaluated with the same ROC curve as before.

Section 03 - Cross Validation on the Diabetes Dataset

1. Implement a 5-fold cross-validation on the diabetes dataset for building a logistic regression model using all feature variables. Obtain 5-fold cross-validated sensitivity, specificity and AUC using the `caret` package.

2. What is the fold with the highest AUC?